

IN THE CLAIMS

Please cancel claims 3, 8, 13-18, 22, 25, 30, and 35-40 without prejudice.

Please amend claims 1-2, 4, 6-7, 9, 11, 19, 23-24, 26, 28-29, 31, 33, 41, and 44 as indicated below.

1. (Currently Amended) A method comprising:

receiving a new policy tree at a network element in a network, wherein the network element stores a current policy tree of classes for quality of service (QoS) of packets being processed by the network element, and wherein the classes of the current policy tree and the classes of the new policy tree include leaf classes and non-leaf classes;

comparing the classes of the current policy tree with the classes of the new policy tree, including

for the current policy tree and the new policy tree, merging, into a set of classification rules of the leaf classes, classification rules of non-leaf classes that are parents of the leaf classes,

identifying a leaf class in the current policy tree as identical to a leaf class in the new policy tree upon determining that the set of classification rules of the leaf class in the current policy tree is equal to the set of classification rules of the leaf class in the new policy tree,

identifying a non-leaf class in the current policy tree as identical to a non-leaf class in the new policy tree upon determining that the non-leaf class in the current policy tree and the non-leaf class in the new policy tree share a greatest number of equivalent descendant leaf classes, and

marking the classes of the current policy tree and the new policy tree as added, deleted, modified or unchanged based on the identifying of the identical leaf and non-leaf classes in the current policy tree and new policy tree; and

selectively deleting classes of the current policy tree based on the comparison of the classes;

verifying each class of the new policy tree with respect to remaining classes of the new policy tree to avoid conflicts when the new policy is merged into a merged policy tree;

associating the merged policy tree with a termination point to activate the merged policy tree on the termination point; and

processing data packets using a packet processing pipeline according to the merged policy tree, the packet processing pipeline including

an incoming packet manager (IPM) to examine a packet header of data packets to determine a next hop of the data packets,

a class identifier (CI) coupled to the IPM to classify the data packets using the merged policy tree,

a route identifier (RI) coupled to the CI to determine which output port through which each of the data packets should be routed,

an outgoing packet manager (OPM) coupled to the RI to store the data packets for outgoing purposes,

a flow identifier (FI) coupled to the OPM to identify one or more flows of which the data packets belong, and

a traffic manager (TM) coupled to the FI to schedule the data packets out of the output port using a result of the FI and the merged policy tree,

wherein each class includes a class name, a type of service, and an amount of bandwidth associated with the respective class, wherein the leaf classes do not have a child class and are orthogonal to a remainder of the leaf classes, and wherein each non-leaf class as a parent class includes at least one leaf class as a child class and each leaf class includes a set of rules that are constrained by a parent class associated with the respective leaf class, wherein a class having a parent class further includes all classification rules included in the parent class, wherein a leaf class as a child of the parent class includes a set of its own rules and attributes and inherits all rules and attributes of its parent class except a root of the respective policy tree, the root representing a data link associated with an output port of the network element, and wherein the rules and attributes of a child class further limit the rules and attributes of its parent.

wherein each of the classes having at least two children classes is identified as a scheduling class and each of the classes not having a child class is identified as a flow class, wherein packets of each flow class are processed in an order of the packets stored in a queue and packets of each scheduling class are processed according to a predetermined schedule, and wherein at least one of the children classes of a scheduling class contains one or more flows that match the respective scheduling class and are not contained in a remainder of the children classes associated with the respective scheduling class.

wherein the network element includes an Ethernet interface card (EIC) coupled to a local area network (LAN) and an ATM interface card (AIC) coupled to a wide area network (WAN), and wherein processing data packets using a packet processing pipeline further comprises

in response to a data packet received at the EIC from the LAN, a CI of the
EIC classifying the data packet using identification of the policy
tree,
an RI of the EIC determining an output port through which the data packet
should be routed using information of the policy tree,
a FI of the AIC determining a flow to which the data packet belongs, using
the policy tree, and
transmitting the data packet to the WAN through the output port according
to QoS based on the policy tree,
wherein the network element further comprises a controller card, wherein the
controller card is to perform the following operations
compiling each policy tree and generate a class lookup table (CLT)
accessible by a CI of the EIC,
associating each policy tree with a termination point and generate a routing
table accessible by each RI for looking up a next hop based on a
class ID, and
creating a list of flow identifier and scheduling update commands to
incrementally change the flow tables for FIs and traffic manager
tables for TMs using flow class and scheduling class property
information of the policy tree,
wherein associating the policy tree with a termination point comprises
creating tables required by the CI and IPM,
differentiating between classes that are currently in service from classes
that will be put into services to generate a list of FI and TM update
commands,

distributing and synchronizing deleted classes by applying the tables and delete commands,
distributing and synchronizing added classes by applying the tables and add commands, and
distributing and synchronizing modified classes by applying the tables and modify commands, and
wherein each class comprises one or more classification rules, QoS requirements, and a classification mask to specify which dimensions are specified in terms of the one or more classification rules, where the classification mask comprises
bit 0 to indicate a source IP address,
bit 1 to indicate a destination IP address,
bit 2 to indicate a source TCP/UDP port,
bit 3 to indicate a destination TCP/UDP port,
bit 4 to indicate an incoming port,
bit 5 to indicate a type of service byte,
bit 6 to indicate a type of protocol used, and
bit 7 to indicate a TCP ACK flag.

2. (Currently Amended) The method of claim 1, wherein the QoS of packets includes a set of parameters which describe required traffic characteristics of a data connection of the packets, including a minimum bandwidth, a maximum delay, a maximum loss and jitter of the data connection, wherein each of the packets includes a packet header having a type of service field to store a value indicating a level of the QoS required for the respective packet, and wherein the level of the QoS is used to identify a class of policy for processing the respective packet by the network element.

3. (Canceled)

4. (Currently Amended) The method of claim 3~~1~~, wherein the selectively deleting classes of the current policy tree comprises deleting a class of the current policy tree upon determining that a set of classification rules of the class of the current policy tree is different than a set of classification rules of a corresponding class of the new policy tree.

5. (Original) The method of claim 4, wherein each class in the current and new policy tree is positioned at a level in the current and new policy tree and wherein the selectively deleting classes of the current policy tree comprises deleting a leaf class of the current policy tree upon determining that that the leaf class of the current policy tree is not positioned at a same level as a leaf class of the new policy tree.

6. (Currently Amended) The method of claim 3~~1~~, wherein the selectively deleting classes of the current policy tree comprises selectively deleting at least one leaf class of the current policy tree.

7. (Currently Amended) The method of claim 3~~1~~, wherein the selectively deleting classes of the current policy tree comprises selectively deleting at least one non-leaf class of the current policy tree.

8. (Canceled)

9. (Currently Amended) The method of claim 3~~1~~, wherein each class is positioned at a level in a policy tree and wherein a leaf class of the current policy tree is identical to a

leaf class of the new policy tree only if the leaf class of the current policy tree and the leaf class of the new policy tree are positioned at an equal level.

10. (Previously Presented) The method of claim 9, wherein each leaf class in the current policy tree and the new policy tree is reciprocally linked to an associated path of non-leaf classes in the current policy tree and new policy tree, respectively, and wherein the selectively deleting the classes of the current policy tree comprises deleting each leaf class in the current policy tree upon determining that the associated path of non-leaf classes in the current policy tree is different from the path of non-leaf classes in the new policy tree for a leaf class.

11. (Currently Amended) The method of claim 31, wherein each class in the current and new policy tree is positioned at a level in the current and new policy tree, wherein each leaf class in the current policy tree and the new policy tree is reciprocally linked to an associated path of non-leaf classes in the current policy tree and new policy tree, respectively, and wherein the selectively deleting classes of the current policy tree comprises deleting a leaf class of the current policy tree upon determining that the associated path of non-leaf classes linked to the leaf class of the current policy tree includes a non-leaf class positioned at a different level than a corresponding non-leaf class included in the associated path of non-leaf classes linked to the leaf class of the new policy tree.

12. (Previously Presented) The method of claim 11, wherein the selectively deleting classes of the current policy tree comprises deleting a leaf class of the current policy tree upon determining that all ancestors of the leaf class of the current policy tree and corresponding ancestors of the leaf of the new policy tree have fewer identical descendant

classes than those had by a class of the current policy tree and a class of the new policy tree positioned at the same level as the parents of the leaf class of the current policy tree and the ancestors of the leaf class of the new policy tree.

13. – 18. (Canceled)

19. (Currently Amended) The method of claim ~~18~~1, wherein each class identifies a subset of packets using one or more classification rules, each classification rule including one or more rule terms, and each rule term including an identity of a data item and a set of constant values associated with the data item, wherein the set of constant values includes at least one of individual values, ranges of constant values, IP subnets expressed in a notation of A.B.C.D/E where A.B.C.D is an IP address and /E indicates a number of leading bits that identify the subnet portion of the IP addresses.

20. (Previously Presented) The method of claim 19, wherein the data item is one of a source IP address, destination IP address, TCP/UDP source port, TCP/UDP destination port, ingress port of the network element, a type of service byte, a protocol type, and TCP acknowledgement flag.

21. (Previously Presented) The method of claim 20, wherein when a data packet is received at the network element, classification rules are evaluated to classify the data packet, wherein if the value of the data item matches any of the constant values associated with a rule term, the rule term is satisfied, wherein if all rule terms of a classification rule are satisfied, the classification rule is satisfied, and wherein the classification rule is satisfied, the data packet is considered to belong to a class associated with the satisfied classification rule.

22. (Canceled)

23. (Currently Amended) A machine-readable medium that provides instructions, which when executed by a machine, cause said machine to perform operations comprising:

receiving a new policy tree at a network element in a network, wherein the network element stores a current policy tree of classes for quality of service (QoS) of packets being processed by the network element, and wherein the classes of the current policy tree and the classes of the new policy tree include leaf classes and non-leaf classes;

comparing the classes of the current policy tree with the classes of the new policy tree, including

for the current policy tree and the new policy tree, merging, into a set of classification rules of the leaf classes, classification rules of non-leaf classes that are parents of the leaf classes,

identifying a leaf class in the current policy tree as identical to a leaf class in the new policy tree upon determining that the set of classification rules of the leaf class in the current policy tree is equal to the set of classification rules of the leaf class in the new policy tree,

identifying a non-leaf class in the current policy tree as identical to a non-leaf class in the new policy tree upon determining that the non-leaf class in the current policy tree and the non-leaf class in the new policy tree share a greatest number of equivalent descendant leaf classes, and

marking the classes of the current policy tree and the new policy tree as added, deleted, modified or unchanged based on the identifying of the identical leaf and non-leaf classes in the current policy tree and new policy tree; and selectively deleting classes of the current policy tree based on the comparison of the classes;

verifying each class of the new policy tree with respect to remaining classes of the new policy tree to avoid conflicts when the new policy is merged into a merged policy tree;

associating the merged policy tree with a termination point to activate the merged policy tree on the termination point; and

processing data packets using a packet processing pipeline according to the merged policy tree, the packet processing pipeline including

an incoming packet manager (IPM) to examine a packet header of data packets to determine a next hop of the data packets,

a class identifier (CI) coupled to the IPM to classify the data packets using the merged policy tree,

a route identifier (RI) coupled to the CI to determine which output port through which each of the data packets should be routed,

an outgoing packet manager (OPM) coupled to the RI to store the data packets for outgoing purposes,

a flow identifier (FI) coupled to the OPM to identify one or more flows of which the data packets belong, and

a traffic manager (TM) coupled to the FI to schedule the data packets out of the output port using a result of the FI and the merged policy tree,

wherein each class includes a class name, a type of service, and an amount of bandwidth associated with the respective class, wherein the leaf classes do

not have a child class and are orthogonal to a remainder of the leaf classes,
and wherein each non-leaf class as a parent class includes at least one leaf
class as a child class and each leaf class includes a set of rules that are
constrained by a parent class associated with the respective leaf class,
wherein a class having a parent class further includes all classification rules
included in the parent class, wherein a leaf class as a child of the parent
class includes a set of its own rules and attributes and inherits all rule and
attributes of its parent class except a root of the respective policy tree, the
root representing a data link associated with an output port of the network
element, and wherein the rules and attributes of a child class further limit
the rules and attributes of its parent,
wherein each of the classes having at least two children classes is identified as a
scheduling class and each of the classes not having a child class is
identified as a flow class, wherein packets of each flow class are processed
in an order of the packets stored in a queue and packets of each scheduling
class are processed according to a predetermined schedule, and wherein at
least one of the children classes of a scheduling class contains one or more
flows that match the respective scheduling class and are not contained in a
remainder of the children classes associated with the respective scheduling
class,
wherein the network element includes an Ethernet interface card (EIC) coupled to
a local area network (LAN) and an ATM interface card (AIC) coupled to a
wide area network (WAN), and wherein processing data packets using a
packet processing pipeline further comprises

in response to a data packet received at the EIC from the LAN, a CI of the
EIC classifying the data packet using identification of the policy
tree,
an RI of the EIC determining an output port through which the data packet
should be routed using information of the policy tree,
a FI of the AIC determining a flow to which the data packet belongs, using
the policy tree, and
transmitting the data packet to the WAN through the output port according
to QoS based on the policy tree,
wherein the network element further comprises a controller card, wherein the
controller card is to perform the following operations
compiling each policy tree and generate a class lookup table (CLT)
accessible by a CI of the EIC,
associating each policy tree with a termination point and generate a routing
table accessible by each RI for looking up a next hop based on a
class ID, and
creating a list of flow identifier and scheduling update commands to
incrementally change the flow tables for FIs and traffic manager
tables for TMs using flow class and scheduling class property
information of the policy tree,
wherein associating the policy tree with a termination point comprises
creating tables required by the CI and IPM,
differentiating between classes that are currently in service from classes
that will be put into services to generate a list of FI and TM update
commands,

distributing and synchronizing deleted classes by applying the tables and delete commands,
distributing and synchronizing added classes by applying the tables and add commands, and
distributing and synchronizing modified classes by applying the tables and modify commands, and
wherein each class comprises one or more classification rules, QoS requirements, and a classification mask to specify which dimensions are specified in terms of the one or more classification rules, where the classification mask comprises
bit 0 to indicate a source IP address,
bit 1 to indicate a destination IP address,
bit 2 to indicate a source TCP/UDP port,
bit 3 to indicate a destination TCP/UDP port,
bit 4 to indicate an incoming port,
bit 5 to indicate a type of service byte,
bit 6 to indicate a type of protocol used, and
bit 7 to indicate a TCP ACK flag.

24. (Currently Amended) The machine-readable medium of claim 23, wherein the QoS of packets includes a set of parameters which describe required traffic characteristics of a data connection of the packets, including a minimum bandwidth, a maximum delay, a maximum loss and jitter of the data connection, wherein each of the packets includes a packet header having a type of service field to store a value indicating a level of the QoS required for the respective packet, and wherein the level of the QoS is used to identify a class of policy for processing the respective packet by the network element.

25. (Canceled)

26. (Currently Amended) The machine-readable medium of claim ~~25~~23, wherein the selectively deleting classes of the current policy tree comprises deleting a class of the current policy tree upon determining that a set of classification rules of the class of the current policy tree is different than a set of classification rules of a corresponding class of the new policy tree.

27. (Original) The machine-readable medium of claim 26 wherein each class in the current and new policy tree is positioned at a level in the current and new policy tree and wherein the selectively deleting classes of the current policy tree comprises deleting a leaf class of the current policy tree upon determining that that the leaf class of the current policy tree is not positioned at a same level as a leaf class of the new policy tree.

28. (Currently Amended) The machine-readable medium of claim ~~25~~23, wherein the selectively deleting classes of the current policy tree comprises selectively deleting at least one leaf class of the current policy tree .

29. (Currently Amended) The machine-readable medium of claim ~~25~~23, wherein the selectively deleting classes of the current policy tree comprises selectively deleting at least one non-leaf class of the current policy tree .

30. (Canceled)

31. (Currently Amended) The machine-readable medium of claim ~~25~~23, wherein each class is positioned at a level in a policy tree and wherein a leaf class of the current policy tree is identical to a leaf class of the new policy tree only if the leaf class of the current policy tree and the leaf class of the new policy tree are positioned at an equal level.

32. (Previously Presented) The machine-readable medium of claim 31, wherein each leaf class in the current policy tree and the new policy tree is reciprocally linked to an associated path of non-leaf classes in the current policy tree and new policy tree, respectively, and wherein the selectively deleting the classes of the current policy tree comprises deleting each leaf class in the current policy tree upon determining that the associated path of non-leaf classes in the current policy tree is different from the path of non-leaf classes in the new policy tree for a leaf class.

33. (Currently Amended) The machine-readable medium of claim ~~25~~23, wherein each class in the current and new policy tree is positioned at a level in the current and new policy tree, wherein each leaf class in the current policy tree and the new policy tree is reciprocally linked to an associated path of non-leaf classes in the current policy tree and new policy tree, respectively, and wherein the selectively deleting classes of the current policy tree comprises deleting a leaf class of the current policy tree upon determining that the associated path of non-leaf classes linked to the leaf class of the current policy tree includes a non-leaf class positioned at a different level than a corresponding non-leaf class included in the associated path of non-leaf classes linked to the leaf class of the new policy tree.

34. (Previously Presented) The machine-readable medium of claim 33, wherein the selectively deleting classes of the current policy tree comprises deleting a leaf class of the

current policy tree upon determining that all parents of the leaf class of the current policy tree and corresponding parents of the leaf of the new policy tree have fewer identical descendant classes than those had by a class of the current policy tree and a class of the new policy tree positioned at the same level as the parents of the leaf class of the current policy tree and the parents of the leaf class of the new policy tree.

35. – 40. (Canceled)

41. (Currently Amended) The machine-readable medium of claim ~~40~~23, wherein each class identifies a subset of packets using one or more classification rules, each classification rule including one or more rule terms, and each rule term including an identity of a data item and a set of constant values associated with the data item, wherein the set of constant values includes at least one of individual values, ranges of constant values, IP subnets expressed in a notation of A.B.C.D/E where A.B.C.D is an IP address and /E indicates a number of leading bits that identify the subnet portion of the IP addresses.

42. (Previously Presented) The machine-readable medium of claim 41, wherein the data item is one of a source IP address, destination IP address, TCP/UDP source port, TCP/UDP destination port, ingress port of the network element, a type of service byte, a protocol type, and TCP acknowledgement flag.

43. (Previously Presented) The machine-readable medium of claim 42, wherein when a data packet is received at the network element, classification rules are evaluated to classify the data packet, wherein if the value of the data item matches any of the constant values associated with a rule term, the rule term is satisfied, wherein if all rule terms of a

classification rule are satisfied, the classification rule is satisfied, and wherein the classification rule is satisfied, the data packet is considered to belong to a class associated with the satisfied classification rule.

44. (Currently Amended) A network element, comprising:
- a processor, and
 - a memory coupled to the processor for storing instructions, when executed from the memory, cause the processor to perform operations including receiving a new policy tree at a network element in a network, wherein the network element stores a current policy tree of classes for quality of service (QoS) of packets being processed by the network element, and wherein the classes of the current policy tree and the classes of the new policy tree include leaf classes and non-leaf classes, comparing the classes of the current policy tree with the classes of the new policy tree, including for the current policy tree and the new policy tree, merging, into a set of classification rules of the leaf classes, classification rules of non-leaf classes that are parents of the leaf classes, identifying a leaf class in the current policy tree as identical to a leaf class in the new policy tree upon determining that the set of classification rules of the leaf class in the current policy tree is equal to the set of classification rules of the leaf class in the new policy tree, identifying a non-leaf class in the current policy tree as identical to a non-leaf class in the new policy tree upon determining that the non-leaf class in the current policy tree and the non-leaf

class in the new policy tree share a greatest number of
equivalent descendant leaf classes, and
marking the classes of the current policy tree and the new policy
tree as added, deleted, modified or unchanged based on the
identifying of the identical leaf and non-leaf classes in the
current policy tree and new policy tree, and
selectively deleting classes of the current policy tree based on the
comparison of the classes,
verifying each class of the new policy tree with respect to remaining classes
of the new policy tree to avoid conflicts when the new policy is
merged into a merged policy tree,
associating the merged policy tree with a termination point to activate the
merged policy tree on the termination point, and
processing data packets using a packet processing pipeline according to the
merged policy tree, the packet processing pipeline including
an incoming packet manager (IPM) to examine a packet header of
data packets to determine a next hop of the data packets,
a class identifier (CI) coupled to the IPM to classify the data packets
using the merged policy tree,
a route identifier (RI) coupled to the CI to determine which output
port through which each of the data packets should be
routed,
an outgoing packet manager (OPM) coupled to the RI to store the
data packets for outgoing purposes,
a flow identifier (FI) coupled to the OPM to identify one or more
flows of which the data packets belong, and

a traffic manager (TM) coupled to the FI to schedule the data packets out of the output port using a result of the FI and the merged policy tree.

wherein each class includes a class name, a type of service, and an amount of bandwidth associated with the respective class, wherein the leaf classes do not have a child class and are orthogonal to a remainder of the leaf classes, and wherein each non-leaf class as a parent class includes at least one leaf class as a child class and each leaf class includes a set of rules that are constrained by a parent class associated with the respective leaf class,

wherein a class having a parent class further includes all classification rules included in the parent class, wherein a leaf class as a child of the parent class includes a set of its own rules and attributes and inherits all rule and attributes of its parent class except a root of the respective policy tree, the root representing a data link associated with an output port of the network element, and wherein the rules and attributes of a child class further limit the rules and attributes of its parent,

wherein each of the classes having at least two children classes is identified as a scheduling class and each of the classes not having a child class is identified as a flow class, wherein packets of each flow class are processed in an order of the packets stored in a queue and packets of each scheduling class are processed according to a predetermined schedule, and wherein at least one of the children classes of a scheduling class contains one or more flows that match the

respective scheduling class and are not contained in a remainder of
the children classes associated with the respective scheduling class.
wherein the network element includes an Ethernet interface card (EIC)
coupled to a local area network (LAN) and an ATM interface card
(AIC) coupled to a wide area network (WAN), and wherein
processing data packets using a packet processing pipeline further
comprises
in response to a data packet received at the EIC from the LAN, a CI
of the EIC classifying the data packet using identification of
the policy tree,
an RI of the EIC determining an output port through which the data
packet should be routed using information of the policy tree,
a FI of the AIC determining a flow to which the data packet
belongs, using the policy tree, and
transmitting the data packet to the WAN through the output port
according to QoS based on the policy tree,
wherein the network element further comprises a controller card, wherein
the controller card is to perform the following operations
compiling each policy tree and generate a class lookup table (CLT)
accessible by a CI of the EIC,
associating each policy tree with a termination point and generate a
routing table accessible by each RI for looking up a next hop
based on a class ID, and
creating a list of flow identifier and scheduling update commands to
incrementally change the flow tables for FIs and traffic

manager tables for TMs using flow class and scheduling
class property information of the policy tree,
wherein associating the policy tree with a termination point comprises
creating tables required by the CI and IPM,
differentiating between classes that are currently in service from
classes that will be put into services to generate a list of FI
and TM update commands,
distributing and synchronizing deleted classes by applying the
tables and delete commands,
distributing and synchronizing added classes by applying the tables
and add commands, and
distributing and synchronizing modified classes by applying the
tables and modify commands, and
wherein each class comprises one or more classification rules, QoS
requirements, and a classification mask to specify which dimensions
are specified in terms of the one or more classification rules, where
the classification mask comprises
bit 0 to indicate a source IP address,
bit 1 to indicate a destination IP address,
bit 2 to indicate a source TCP/UDP port,
bit 3 to indicate a destination TCP/UDP port,
bit 4 to indicate an incoming port,
bit 5 to indicate a type of service byte,
bit 6 to indicate a type of protocol used, and
bit 7 to indicate a TCP ACK flag.